

Back End Engineering-II

Project Report

Semester-V (Batch-2023)

RentEase



Supervised by:

Dr. Shivani Wadhwa

Submitted By:

Maridul Walia (2310991897)

Mehak Walia (2310991901)

Ramya Padala (2310992587)

**Department of Computer Science and Engineering Chitkara
University Institute of Engineering and Technology, Chitkara
University, Punjab**

Abstract

The evolution of digital ecosystems has led to a significant paradigm shift from traditional ownership models toward access-based consumption. This transformation has been largely influenced by the rapid growth of the sharing economy, where individuals prefer temporary access to products and services rather than permanent ownership. In this context, peer-to-peer rental platforms have emerged as powerful tools that enable users to utilize underused assets, reduce financial burdens, and promote sustainable consumption practices.

RentEase is developed as a full-stack rental management web application that addresses the growing demand for streamlined rental operations. The platform is built using the MERN stack—MongoDB, Express.js, React.js, and Node.js—allowing it to deliver a highly responsive, scalable, and flexible solution. RentEase enables users to list items they own, explore items available for rent, and manage rental activities through an integrated, user-friendly dashboard. The platform ensures ease of navigation and smooth interaction by leveraging React's component-based architecture and Node.js' non-blocking backend processing.

To maintain reliability and trust between users, the system incorporates structured workflows, including mandatory user verification, detailed item listing procedures, secure booking mechanisms, and complete transaction tracking. Owners can define item availability, rental pricing, and security deposit values, while renters can browse items through category filters, view detailed descriptions, and initiate booking requests. Furthermore, the platform includes a dedicated administrator module responsible for overseeing the entire application ecosystem. Admins can verify user documents, monitor suspicious activities, suspend accounts if required, and manage user complaints to ensure a safe and transparent environment.

At this stage of development, the application uses local file storage to handle user-uploaded documents and images such as identity proofs, profile photos, and item pictures. However, the architectural design includes provisions to integrate cloud-based storage services like AWS S3, Cloudinary, or Firebase Storage in future iterations to enhance scalability, accessibility, and reliability. While deployment to cloud hosting services and full version control practices were not implemented in the current project phase, these elements are part of the planned development pipeline for upcoming releases.

Overall, the results of this development phase demonstrate that RentEase is capable of effectively managing rental operations by reducing manual errors, improving user transparency, and creating a structured workflow for both renters and owners. The modular nature of the MERN stack ensures that the platform can be easily extended with advanced functionalities in the future, including payment gateway integration, automated refund processing, real-time notifications, and enhanced security deposit automation. Thus, the project establishes a strong foundation for a scalable, dependable, and user-centric rental management system suited for real-world deployment.

Tables of Content

S.NO	CONTENT	PAGE NO.
1.	Introduction	3 - 5
2.	Problem Statement & Requirements	6 - 7
3.	System Implementation	8 - 14
4.	Result	15 - 29
5.	Conclusion	30
6.	References	31

1. Introduction

1.1. Background

In recent years, the global marketplace has experienced a transformative shift due to the evolution of digital platforms and the rapid rise of the sharing economy. This new economic paradigm emphasizes access over ownership, enabling individuals to temporarily use items instead of buying them outright. The model gained prominence with the emergence of major platforms facilitating shared transportation, accommodation, and professional services. As individuals become more conscious of financial efficiency, convenience, and sustainability, the appeal of renting items that are used occasionally has increased substantially. Whether it is a household appliance, a piece of musical equipment, a vehicle, or specialized tools, renting is often more economical and environmentally responsible than ownership.

The sharing economy is driven by advancements in networking technologies, secure digital transactions, and online identity systems. However, despite these improvements, many peer-to-peer item rentals still function informally. Traditional methods heavily rely on interpersonal trust, verbal commitments, and manual coordination through phone calls or social media. This leads to several recurring problems—users cannot verify the reliability of strangers, item owners have no assurance that their property will be returned safely, and renters may face inconsistent pricing or availability. These inefficiencies discourage many potential users from participating in peer-to-peer rentals.

Another key challenge is the absence of a centralized management system capable of organizing the entire rental lifecycle. Without a digital record of listings, bookings, pickup times, rental periods, payments, and disputes, misunderstandings become common. For instance, an item may be double-booked, returned late, or found damaged without clear accountability. Similarly, owners may not have an easy way to track earnings or manage multiple items, while renters may lack visibility into their pending or past bookings. These barriers highlight the necessity of a structured, transparent, and technologically supported rental process.

To address these issues, RentEase has been conceptualized and developed as a modern digital rental management system. Built using the MERN stack (MongoDB, Express.js, React.js, and Node.js), RentEase offers a unified platform that brings together owners, renters, and administrators in a secure and efficient system. The platform leverages the strengths of its underlying technologies—MongoDB's flexible document storage, Node.js and Express.js for robust backend logic, and React.js for a dynamic, user-friendly interface.

RentEase positions itself as a comprehensive solution to facilitate trustworthy item rentals. It includes features such as user verification through ID uploads, detailed item listings with images and availability, request-based booking, security deposit handling, and multi-stage booking workflows. By systematizing every aspect of the rental process, RentEase not only resolves the common issues encountered in traditional peer-to-peer rentals but also encourages more users to participate confidently.

Additionally, the digitalization of the rental ecosystem aligns with modern sustainable practices. Renting instead of purchasing reduces the demand for mass production and helps individuals utilize resources more responsibly. As environmental concerns grow, systems like RentEase contribute to lowering carbon footprints by extending the useful life of existing items. Thus, the platform is not merely a technological product but also a social enabler and a step toward a circular economy.

1.2.Objectives

- The development of RentEase is guided by a set of well-defined objectives aimed at delivering a structured, scalable, and user-centric rental management system. These objectives encompass technical goals, user experience improvements, administrative oversight, and long-term adaptability.

1.2.1. Technical Objectives

- To build a full-stack web application using the MERN stack, ensuring modern development practices, a responsive user interface, and high backend performance.
- To design a scalable and modular architecture capable of supporting additional features such as cloud deployment, payment gateways, and automated workflows.
- To implement secure API endpoints for handling authentication, item management, bookings, payments, and administrative actions.
- To develop a flexible database schema using MongoDB that can efficiently store diverse item categories, transaction records, user profiles, and rental histories.

1.2.2. User-Centric Objectives

- To provide users with the ability to list items easily, including detailed specifications such as images, rental rates, available dates, condition rating, security deposits, and location.
- To offer renters an intuitive browsing experience through search and filter functionalities (category, cost, condition, location, availability).
- To support a complete rental workflow, including booking requests, approval, pickup confirmation, active rental periods, returns, and reviews.
- To facilitate effective communication and transparency through clear status indicators for bookings (Pending, Approved, Active, Completed).
- To deliver a comprehensive user dashboard displaying wallet balance, earnings, rental history, listed items, and bookings—all in one integrated interface.

1.2.3. Administrative Objectives

- To implement an administrator panel with functionalities such as viewing all users, verifying accounts, suspending accounts, handling complaints, regulating platform fee settings, and monitoring system activity.
- To maintain platform integrity through structured user verification using uploaded identity documents.

- To enable configuration of system-wide settings, including platform fees, referral bonuses, minimum rental days, maximum rental limits, and auto-approval functionalities.
- To provide audit and monitoring capabilities, giving admins visibility into revenue generation, booking trends, and user activity patterns.

1.2.4. Long-Term Objectives

- To prepare the system for cloud-based deployment, enabling scalability, availability, and seamless user experience.
- To plan integration of cloud storage services (AWS S3, Cloudinary, Firebase Storage) for handling user-uploaded documents and images.
- To support future enhancements, such as real-time notifications, chat systems, payment gateway integration, and automated deposit settlement.
- These objectives collectively ensure that RentEase functions as a complete and dependable rental ecosystem capable of adapting to future needs.

1.3. Significance

The significance of RentEase extends beyond being just a software project; it has broader implications in technological, economic, and societal contexts like: -

- Efficient handling of real-time data
- Component-based UI architecture
- Asynchronous backend processes
- Flexible data storage models
- Secure authentication and authorization workflows
- It serves as a practical example for students and developers exploring advanced web technologies and scalable application design.
- Earn additional income
- Access expensive or specialized tools without purchase
- Share underutilized resources

2. Problem Statement & Requirements

2.1. Problem Statement

The rise of the sharing economy has transformed traditional consumption patterns by enabling individuals to access goods and services without the requirement of permanent ownership. Despite this shift, existing peer-to-peer item rental processes often remain informal, fragmented, and inefficient. Many individuals who wish to rent out their possessions lack a reliable system to manage listings, communicate with potential renters, track rental timelines, or verify user authenticity. On the other hand, users seeking short-term access to items often struggle to locate trustworthy owners, evaluate item quality, ensure availability, or maintain transparent communication during the rental lifecycle. These limitations collectively restrict the growth of a fully functional peer-to-peer rental ecosystem.

A detailed analysis of current rental practices reveals several critical shortcomings:

2.1.1. Lack of a Unified and Structured Platform

Most peer-to-peer rentals occur through informal channels such as social media groups, messaging apps, or verbal agreements. These channels do not provide standardization, reliable documentation, or structured workflows. As a result, users face inconsistencies in pricing, item availability, and rental terms.

2.1.2. Absence of User Verification and Trust Mechanisms

Trust remains the biggest barrier in peer-to-peer rental environments. Without identity verification, renters may fail to return items on time or may misuse them, while owners may cancel bookings unexpectedly. This absence of accountability discourages participation and increases the perceived risk of item damage, loss, or financial disputes.

2.1.3. Poor Coordination of Rental Activities

Traditional rental interactions depend heavily on manual coordination—phone calls, text messages, and face-to-face discussions. This approach often results in confusion about rental dates, item availability, pickup timing, or return deadlines. Human error can lead to overlapping bookings, last-minute cancellations, or denied requests.

2.1.4. No Transparent Record-Keeping or Tracking

In manual rental systems, there is no structured method to maintain records of transactions, user interactions, or item history. Users are unable to track past rentals, financial details, security deposits, or dispute resolutions. This makes it difficult to hold parties accountable and complicates administrative oversight.

2.1.5. Lack of Security Deposit and Payment Structure

Many peer-to-peer rentals happen without proper security deposits or predefined payment terms. Without a formal mechanism to ensure item safety or compensate for damages, owners are hesitant to list valuable items. Renters also lack clarity on deposit amounts,

refund conditions, and rental pricing standards.

2.1.6. No Central Administration or Complaint Resolution Framework

Without a supervisory entity or administrative panel, disputes cannot be efficiently managed. Complaints about item damage, late returns, or fraudulent behavior often go unresolved. This hampers user confidence and prevents long-term adoption of rental platforms.

2.1.7. Limited Digitalization of Rental Lifecycle

The modern user experience requires automation, real-time updates, and intuitive interfaces—features that traditional rental processes completely lack. A digitalized system should support searchability, booking workflows, notifications, and status tracking, none of which are available in informal arrangements.

2.2. Requirements

2.2.1. Software Requirements

- Operating System: - Windows 10/11.
- Backend Technologies: - Node.js (v16 or above), express.js, MongoDB(Atlas or Compass), Mongoose ODM
- Frontend Technologies: - React.js
- Development Tools: - VS Code, Postman, Git, npm/yarn

2.2.2. Hardware Requirements

- Processor: - Intel Core i5(8th Gen or above) or above, AMD Ryzen 5 or above
- RAM: - Minimum 8 GB
- Storage: - Minimum 20 GB free space

3. System Implementation

The implementation of the RentEase platform involves the integration of multiple technologies, development strategies, architectural decisions, and system components that together form a comprehensive peer-to-peer rental management system. This section outlines the overall system implementation in detail, covering the technology stack, software and hardware requirements, architecture design, module-wise implementation, data flow, and operational workflow.

RentEase is built using the MERN stack (MongoDB, Express.js, React.js, Node.js), chosen for its scalability, ease of development, and seamless integration between frontend and backend components. The platform supports multiple user roles, including general users (renters and owners) and an administrator, each with structured workflows that guide them through the rental lifecycle—from item listing to booking, approval, pickup, return, and review.

3.1. Technology Stack

The MERN stack forms the core of the RentEase platform:

3.1.1. MongoDB (Database Layer)

A NoSQL database used to store structured data such as user profiles, items, bookings, complaints, reviews, and system configurations. Its flexible schema allows dynamic objects such as item details and transaction histories to be stored efficiently.

3.1.2. Express.js (Backend Framework)

Used to develop RESTful APIs for user authentication, item listing, booking management, wallet transactions, and administrative operations. Express simplifies server-side logic, middleware handling, and data routing.

3.1.3. React.js (Frontend Framework)

Provides a component-based UI for the platform's dashboards, forms, navigation menus, item listings, and admin interface. React ensures fast, interactive, and smooth user experiences.

3.1.4. Node.js (Server Runtime)

Handles server operations, executes business logic, manages API responses, authenticates requests, and communicates with the database. Node.js ensures non-blocking I/O and high concurrency.

3.2. System Architecture

The architecture of RentEase follows a three-tier model combining:

3.2.1. Presentation Layer (Frontend UI)

- Built using React.js
- Includes dashboard, item listings, booking interface, admin panel
- Communicates with backend via HTTPS API calls

3.2.2. Application Layer (Backend Server)

- Node.js with Express.js
- Implements routing, input validation, business logic
- Handles authentication, booking workflow, deposit calculations

3.2.3. Data Layer (MongoDB Database)

- Stores users, items, bookings, complaints, reviews, and admin configurations
- Uses Mongoose models to enforce schema rules

3.2.4. File Handling Module

- Uses Multer to upload:
- User ID proof images
- Item images
- Stored in local filesystem with future cloud integration

3.3. System Modules Implementation

The RentEase platform is divided into multiple functional modules:

3.3.1. User Management Module

Functions:

- User registration with name, email, phone number, address, ID proof, password
- Login via email and password
- JWT-based authentication
- User dashboard access
- Profile updates (name, phone, address, profile image)
- Change password option
- Viewing wallet balance and transaction history

Verification Flow:

- Admin verifies users through the admin panel
- Verification status is displayed in profile settings
- Unverified users may have restricted actions

3.3.2. Item Listing Module

Users can list items for rent with a detailed form that includes:

- Item name
- Description
- Category (Appliances, Vehicles, Tools, Sports, Instruments, etc.)
- Condition (Excellent, Good, Fair, Poor)
- Daily rental price
- Item value
- Security deposit percentage (10%, 20%, 30%, 50%)
- Location
- Availability dates (from-to)
- Multiple images

Backend logic includes:

- Image storage
- Availability validation

- Category filtering
- Rental value and deposit calculations

3.3.3. Booking and Rental Workflow Module

The rental lifecycle follows a structured sequence:

- Pending Request
A renter sends a booking request → item owner receives notification.
- Approved Request
Owner approves booking → item is reserved.
- Picked Up Status
Renter collects item → status updated to Active.
- Return and Completion
Item is returned → transaction moves to Completed.
- Reviews & Ratings
Users give ratings and feedback after completion.
- D. Wallet and Payment Tracking Module
Although full payment gateway integration is future scope, basic wallet operations exist:
 - Manual wallet top-up
 - Security deposit tracking
 - Rental cost logs
 - Transaction history (credit/debit)

3.3.4. Search and Filter Module

The Items page supports:

- Keyword search
- Category filters
- Minimum and maximum price filters
- Location-based filtering
- Sorting (newest first)
- Pagination (page 1, page 2, or view all)

The backend uses MongoDB queries with regex and filter operators.

3.3.5. Admin Panel Module

The admin account has exclusive functionality:

Dashboard View:

- Total users
- Total items
- Platform revenue
- Active bookings
- Recent bookings
- Complaints overview

User Management:

- View all users
- Verify accounts
- Suspend accounts
- Add verification message

Complaints Module:

- View, filter, and resolve user complaints

Reviews Module:

- View all platform reviews
- Identify spam or inappropriate feedback

Admin Settings:

- Configure platform fee
- New user bonus
- Referral bonus
- Maximum items per user
- Rental settings (min/max rental days)
- Auto-approve bookings toggle

3.4. API Architecture

RentEase uses RESTful APIs for communication between client and server. The backend exposes endpoints categorized as follows:

3.4.1. Auth Routes (/api/auth)

Method	Path	Auth	Purpose
POST	/api/auth/register	No	Register a user
POST	/api/auth/login	No	Login & get JWT
GET	/api/auth/profile	Yes	Get profile of logged-in user
PUT	/api/auth/profile	Yes	Update profile (optional profileImage upload)
PUT	/api/auth/change-password	Yes	Change password

POST	/api/auth/add-to-wallet		Add money to wallet (amount)
------	-------------------------	--	------------------------------

3.4.2. Item Routes (/api/auth)

Method	Path	Auth	Purpose
GET	/api/items	No	Browse items with query filters
GET	/api/items/categories	No	Get categories and counts
GET	/api/items/:id	No	Get item detail
POST	/api/items	Yes	Create item (images upload)
PUT	/api/items/:id	Yes	Update item (owner only)
DELETE	/api/items/:id	Yes	Delete item (owner only)
GET	/api/items/user/my-items	Yes	Get items of current user
POST	/api/items/extend-rental	Yes	Request extension (extendRentalPeriod)

3.4.3. Booking Routes (/api/bookings)

Method	Path	Auth	Purpose
POST	/api/bookings	Yes	Create new booking (itemId, startDate, endDate)
GET	/api/bookings	Yes	Get user bookings (borrower/lender/all, status filter)
GET	/api/bookings/:id	Yes	Get booking detail (authorization check)
PUT	/api/bookings/:id/status	Yes	Update booking status (approved/cancelled/active/completed)
POST	/api/bookings/:id/messages	Yes	Add message to booking conversation

3.4.4. Review Routes (/api/reviews)

Method	Path	Auth	Purpose
GET	/api/reviews/item/:itemId	No	Get reviews for an item (paginated)
GET	/api/reviews/user/:userId	No	Get reviews about a user
POST	/api/reviews	Yes	Create a review for a completed booking
GET	/api/reviews/my-reviews	Yes	Get reviews written by current user

POST	/api/reviews/:id/report	Yes	Report a review (reason)
------	-------------------------	-----	--------------------------

3.4.5. Complaint Reviews (/api/complaints)

Method	Path	Auth	Purpose
POST	/api/complaints	Yes	File complaint (upload evidence)
GET	/api/complaints	Yes	Get user's complaints (filed/received/all)
GET	/api/complaints/:id	Yes	Get complaint details (authorization check)
POST	/api/complaints/:id/messages	Yes	Add message to complaint

3.4.6. Admin Routes (/api/admin)

Method	Path	Auth	Purpose
GET	/api/admin/maintenance-status	Public	Public endpoint showing maintenance info
GET	/api/admin/dashboard	Admin	Get admin dashboard stats (users/items/bookings/complaints/recent)
GET	/api/admin/users	Admin	List users with filters & pagination
GET	/api/admin/users/:id	Admin	Get user details with recent activity
PUT	/api/admin/users/:id/suspend	Admin	Suspend / unsuspend user (reason)
PUT	/api/admin/users/:id/verify	Admin	Toggle user verification
GET	/api/admin/complaints	Admin	Get all complaints
PUT	/api/admin/complaints/:id	Admin	Update complaint (status, resolution)
GET	/api/admin/reviews/reported	Admin	Get reported reviews
PUT	/api/admin/reviews/:id/toggle-visibility	Admin	Hide/unhide review
GET	/api/admin/settings	Admin	Get settings (singleton)
PUT	/api/admin/settings	Admin	Update settings (maintenance mode handling included)

3.5. Data Flow Overview

3.5.1. Registration Flow

User submits form → validation → password hashing → MongoDB entry → pending verification.

3.5.2. Login Flow

User enters credentials → server verifies hashed password → JWT token generated.

3.5.3. Booking Flow

Search item → send request → owner approves → update booking status → pickup → return → review.

3.5.4. Admin Flow

Views user → verifies → updates system configs → resolves complaints.

3.6. System Security Implementation

- JWT-based authentication
- Password hashing using bcrypt
- Authorization middleware
- File validation (image format, size)
- Role-based access (user vs admin)

4. Results

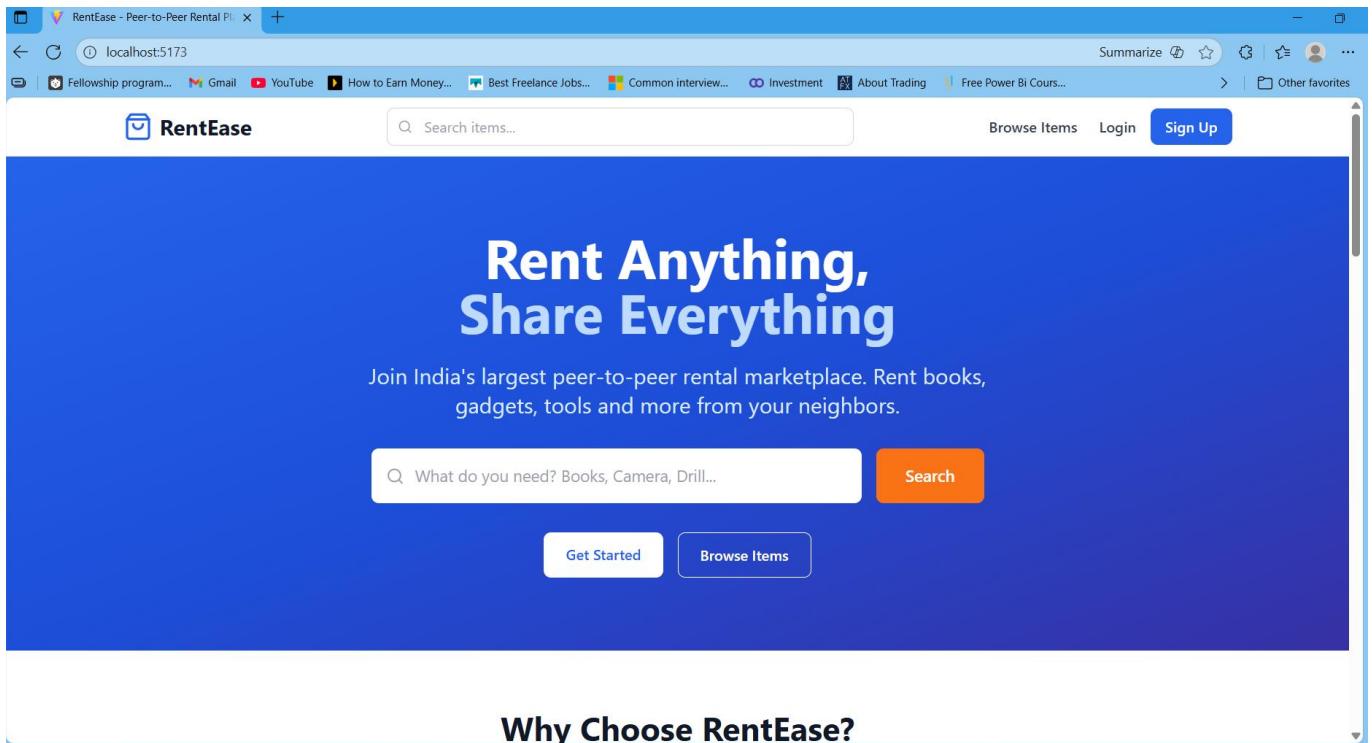
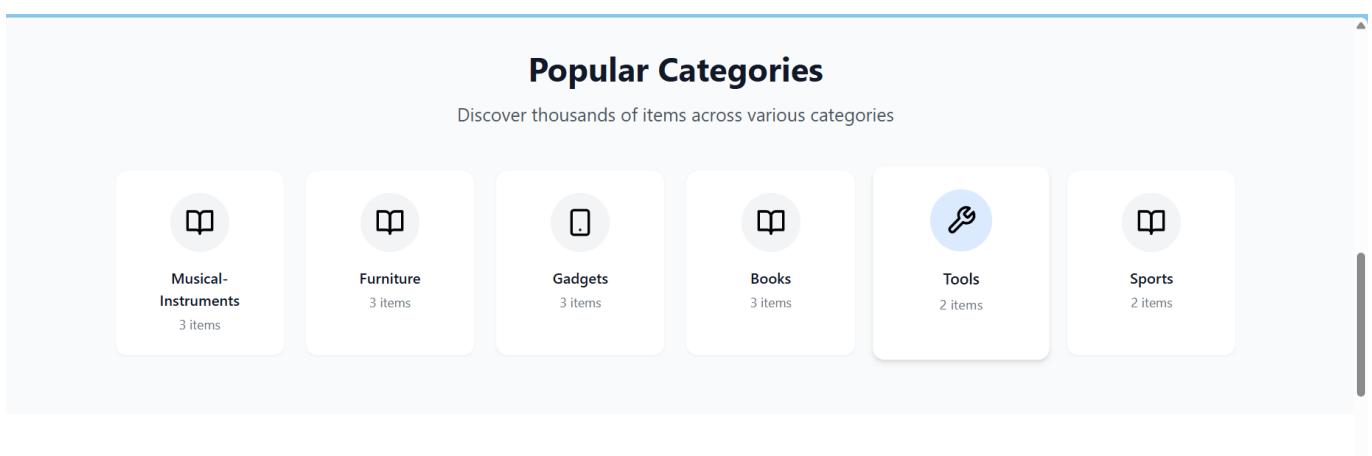


Fig 1 – Home page



Featured Items

Top-rated items from trusted lenders

[View All >](#)



Fig 2 – Home page (Popular Categories)

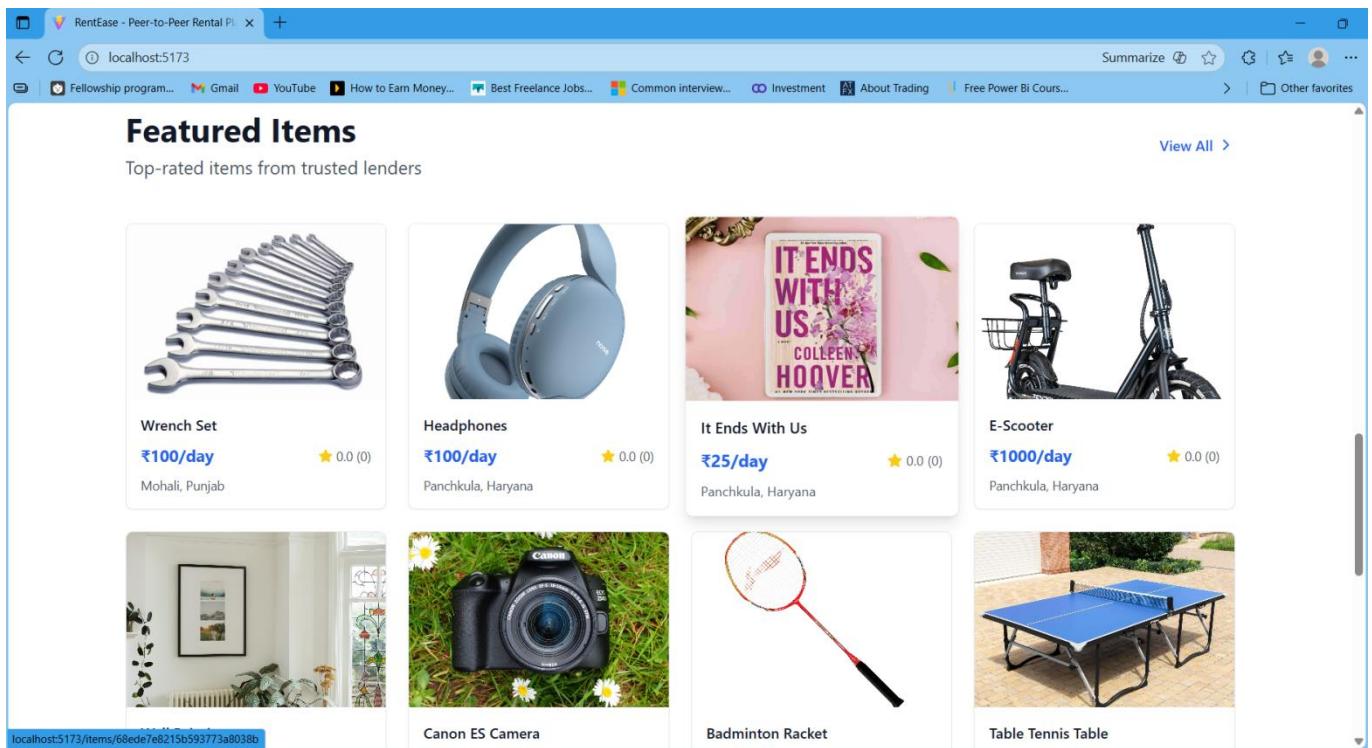


Fig 3 – Home page (Featured Items)

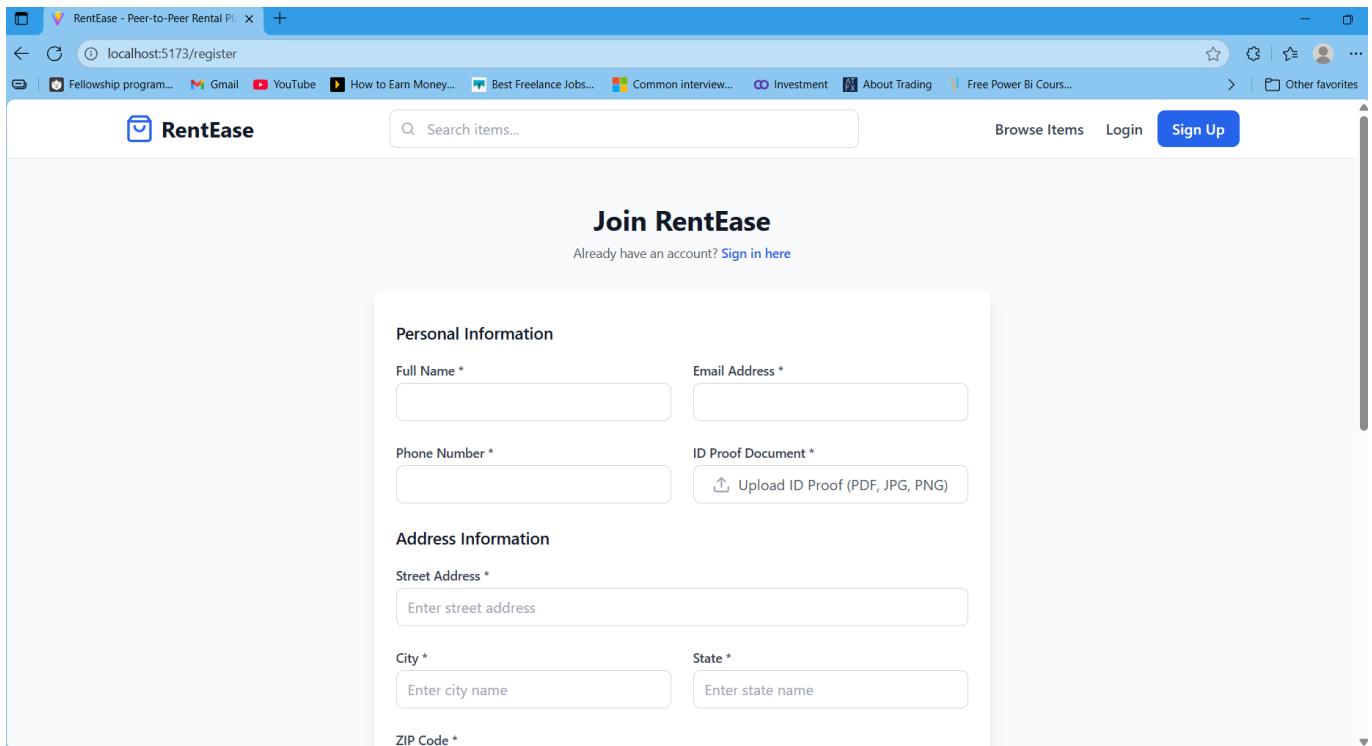


Fig 4 – Signup page

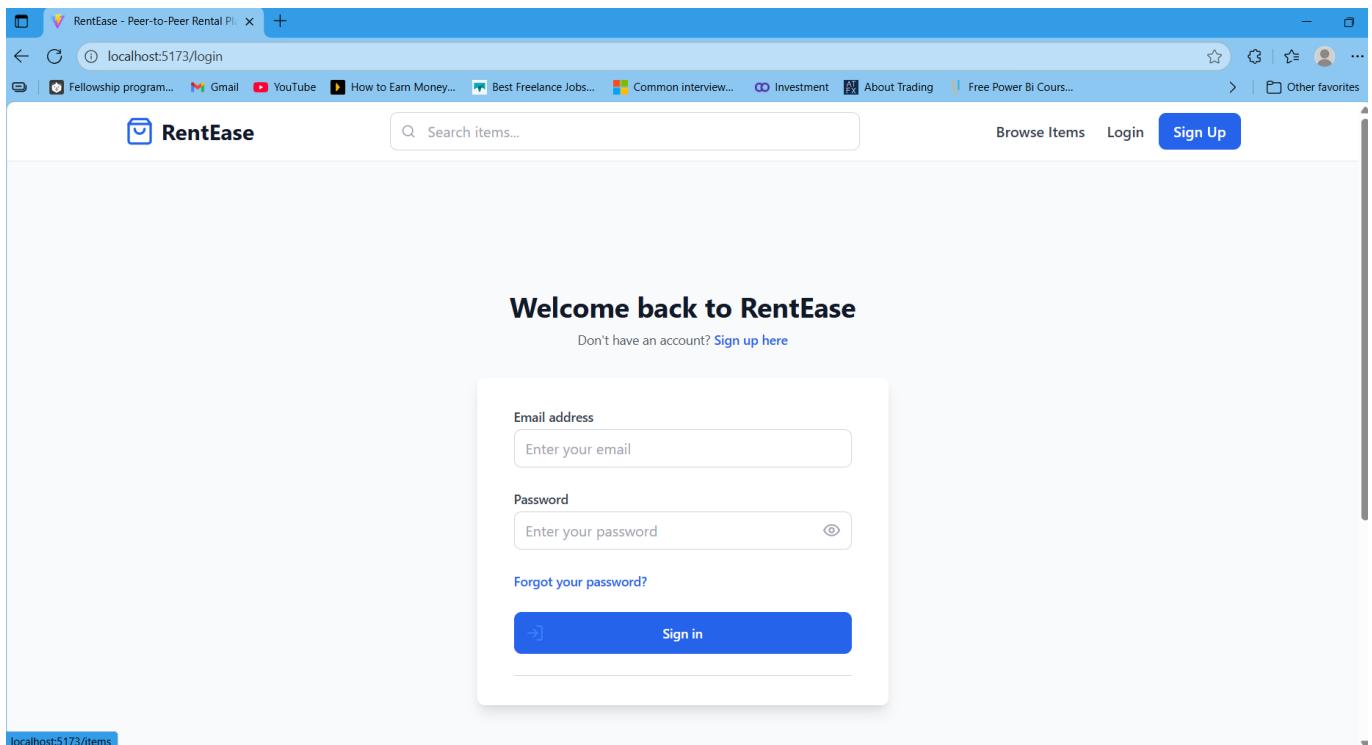


Fig 5 – Login page

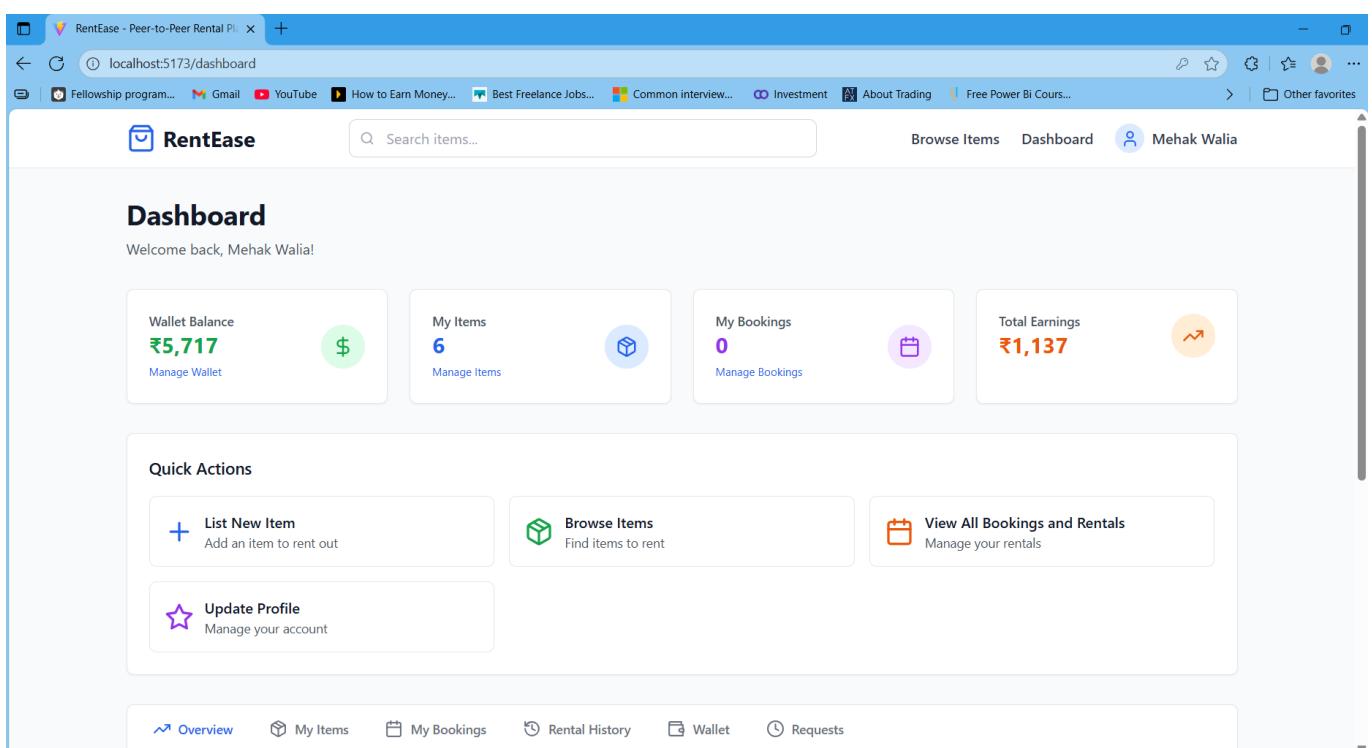


Fig 6 – Dashboard page

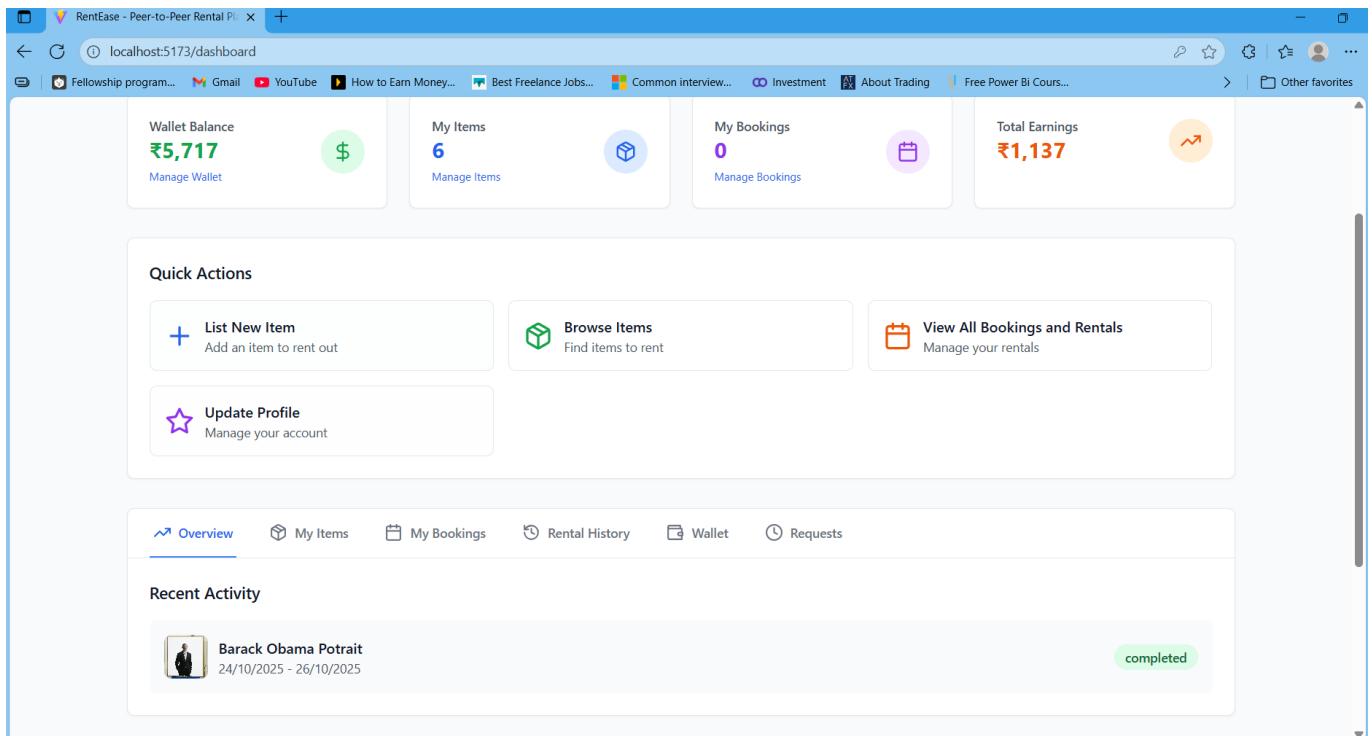


Fig 7 – Dashboard page (Quick Actions & Tabs section – Overview Tab)

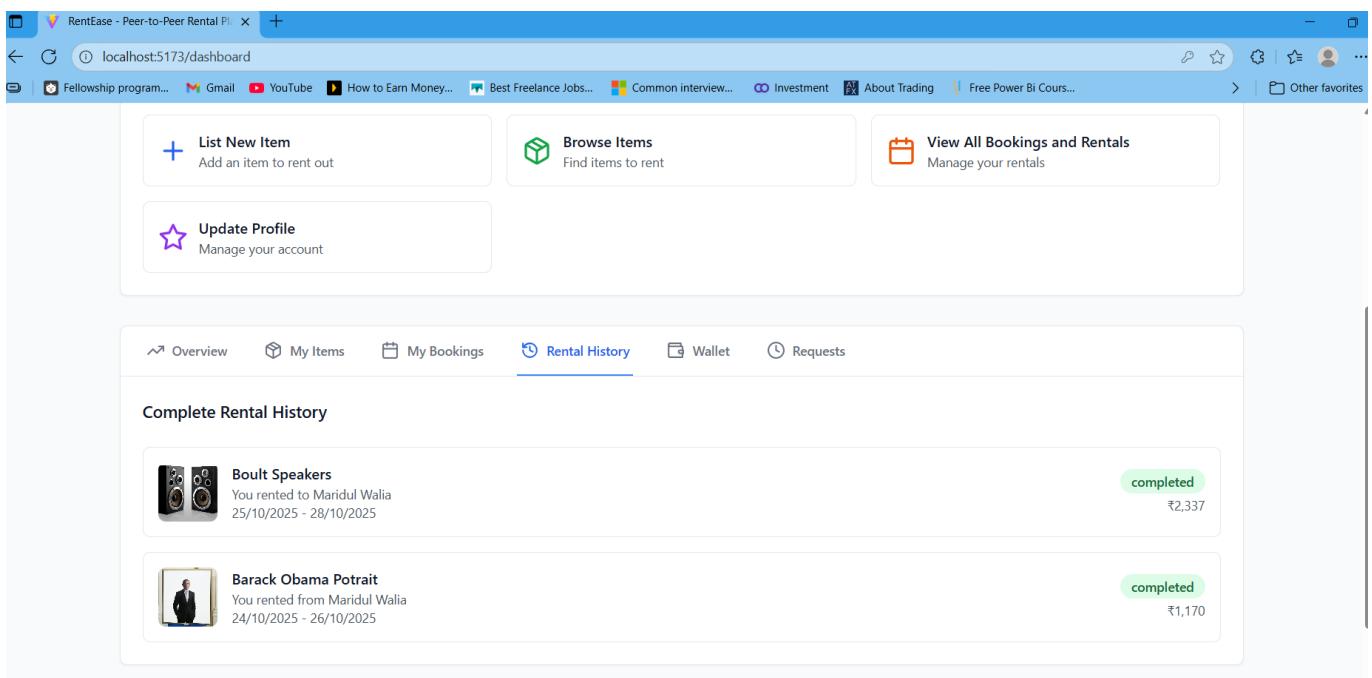


Fig 8 – Dashboard page (Quick Actions & Tabs section – Rental History Tab)

The screenshot shows the RentEase dashboard with the 'Wallet' tab selected. At the top, there's a 'Quick Actions' section with a purple star icon and the text 'Update Profile Manage your account'. Below the tabs, it says 'Wallet Management'. The 'Add Money to Wallet' section has a text input placeholder 'Enter amount' and a grey button 'Add to Wallet'. To the right, the 'Recent Transactions' section lists four items:

- Earnings from rental: Bolt Speakers 23/10/2025 +₹1137
- Deposit refund for: Barack Obama Potrait 23/10/2025 +₹750
- Booking payment for Barack Obama Potrait 23/10/2025 -₹1170
- Wallet top-up of ₹4000 23/10/2025 +₹4000

Fig 9 - Dashboard page (Quick Actions & Tabs section – Wallet Tab)

The screenshot shows the RentEase dashboard with the 'My Bookings' tab selected. At the top, there's a 'Quick Actions' section with three buttons: '+ List New Item Add an item to rent out', 'Browse Items Find items to rent', and 'View All Bookings and Rentals Manage your rentals'. Below the tabs, it says 'My Bookings (1)'. A single booking is listed with a blue 'View All Bookings' button:

	Barack Obama Potrait 24/10/2025 - 26/10/2025 Owner: Maridul Walia	completed ₹1,170
--	---	---

Fig 10 - Dashboard page (Quick Actions & Tabs section – My Bookings Tab)

The screenshot shows the RentEase Peer-to-Peer Rental Platform dashboard. At the top, there's a blue header bar with the title 'RentEase - Peer-to-Peer Rental Pl...' and a URL 'localhost:5173/dashboard'. Below the header is a navigation bar with links to various websites like Fellowship program..., Gmail, YouTube, etc. The main content area has a 'Quick Actions' section with three buttons: 'List New Item' (Add an item to rent out), 'Browse Items' (Find items to rent), and 'View All Bookings and Rentals' (Manage your rentals). Below this is a tabs section with 'Overview', 'My Items', 'My Bookings' (which is currently selected), 'Rental History', 'Wallet', and 'Requests'. The 'My Bookings' tab displays a message 'Booking Requests (0)' and a 'No pending requests' message with a clock icon, indicating that new booking requests will appear here.

Fig 11 - Dashboard page (Quick Actions & Tabs section – My Bookings Tab)

The screenshot shows the RentEase Peer-to-Peer Rental Platform dashboard. The 'My Items' tab is selected, indicated by a blue underline. The top navigation bar and quick actions are identical to Fig 11. The main content area shows a list of six items: 'Wooden Utensils' (₹649/day, 0.0 rating, Available), 'Acer Aspire Lite' (₹5000/day, 0.0 rating, Available), 'Trumpet' (₹367/day, 0.0 rating, Available), 'Speaker' (not visible in the screenshot), 'Lamp' (not visible in the screenshot), and 'Guitar' (not visible in the screenshot). Each item has a small thumbnail image, its name, price per day, average rating (0.0 (0)), and an 'Available' status indicator. There is also a '+ Add Item' button in the top right corner of the item list.

Fig 12 - Dashboard page (Quick Actions & Tabs section – My Items Tab)

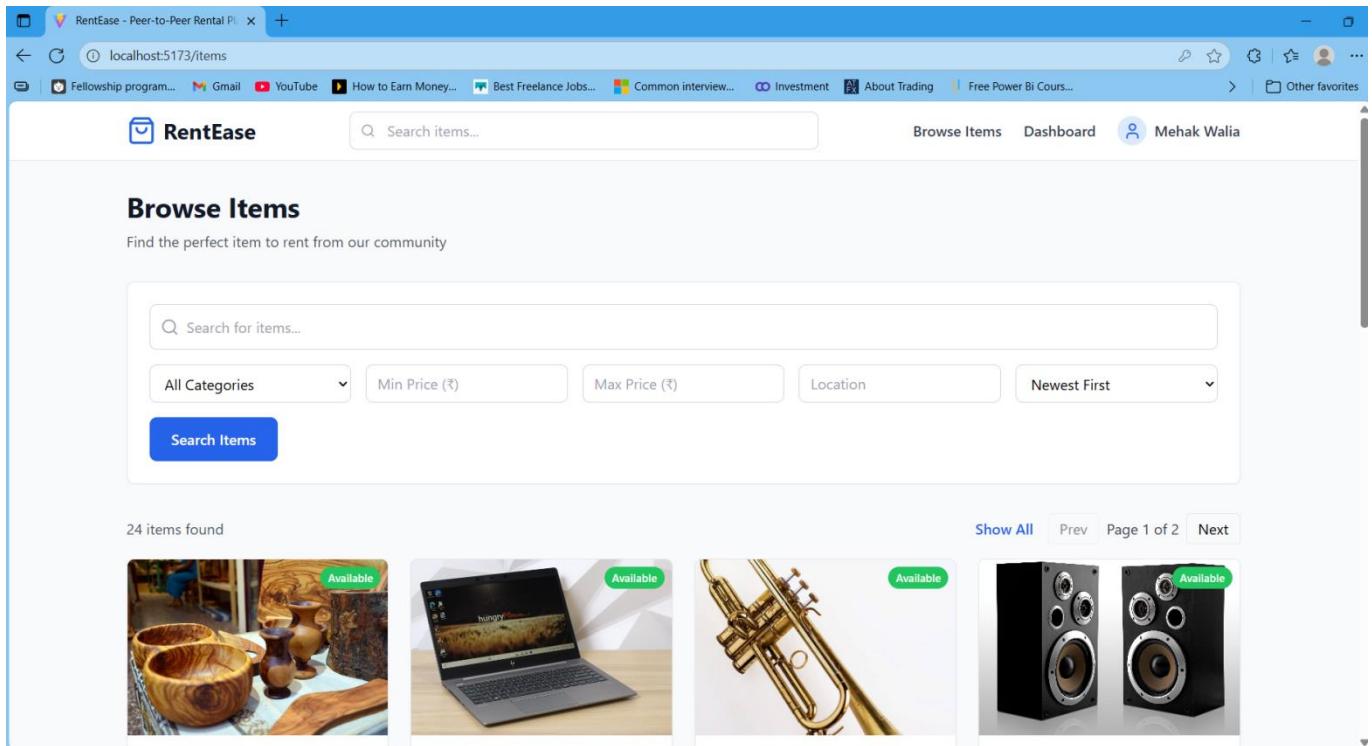


Fig 13 – Browse Items

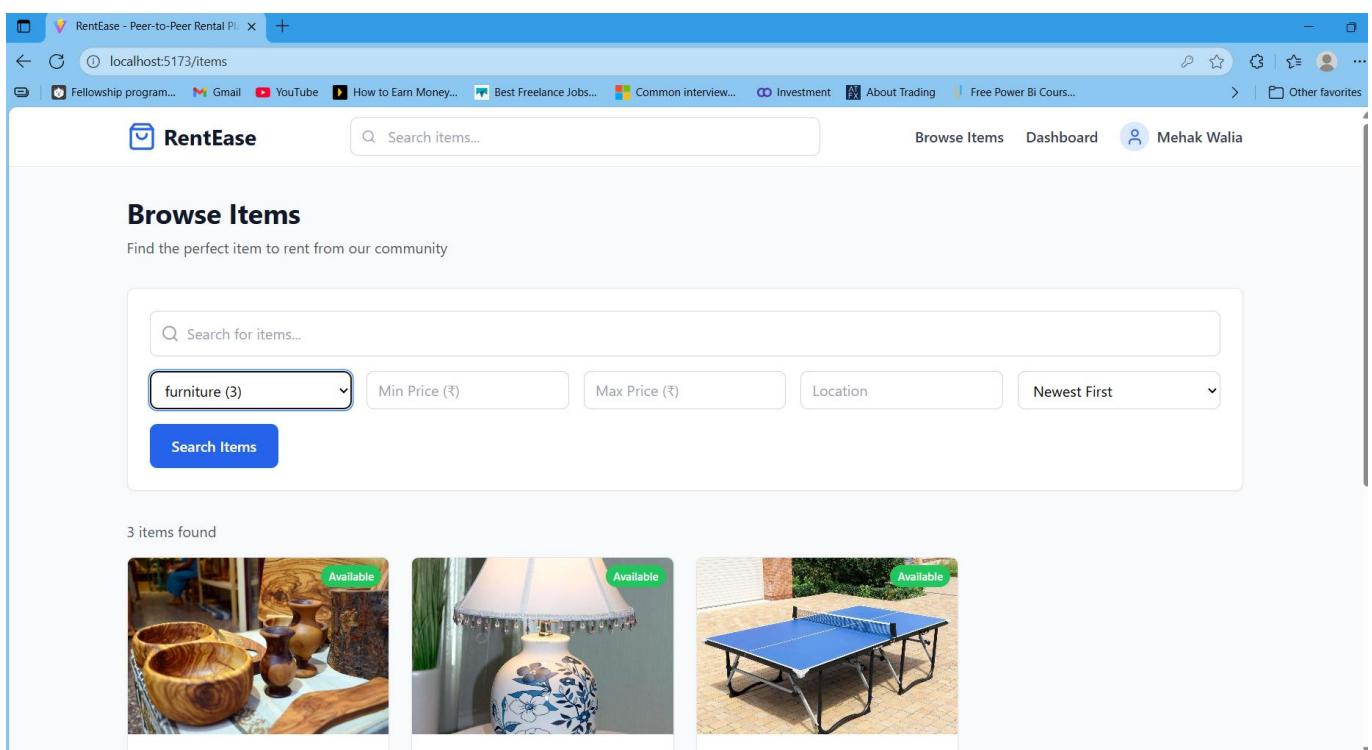


Fig 14 – Browse Items (Filtered through Categories)

The screenshot shows a web browser window for the RentEase platform. The URL is localhost:5173/items/68f92c180c01a1011aa9c038. The page title is "RentEase - Peer-to-Peer Rental Pl...". The main content is a listing for a "Cricket Bat". The bat is shown in four different views. The title "Cricket Bat" is bolded. Below it is a yellow star icon with "0.0 (0 reviews)" and a location pin with "Chandigarh, Chandigarh". The price "₹199 per day" is displayed in blue, with a green "Available" button to its right. A "Description" section follows, stating "bat has proper stroke with leather ball". Below this are two tables showing "Category: Sports", "Item Value: ₹1,800", "Security Deposit: ₹540", and "Condition: Fair". The "Owner Information" section shows a profile picture of Maridul Walia, a yellow star icon with "0.0 (0 reviews)", and contact icons for phone and email.

Fig 15 – Items Details Page

This screenshot shows the same RentEase items details page as Fig 15, but with a focus on the "Book This Item" section. The "Start Date" and "End Date" fields are present, both with "dd-mm-yyyy" placeholder text and calendar icons. Below these is a "Message to Owner (Optional)" field containing the placeholder "Tell the owner about your rental needs...". A large blue "Send Booking Request" button is centered at the bottom of this section. Below this section is a "Reviews" section, which includes a "Your Rating" dropdown set to "5", a text area for "Share your experience with this item...", and a "Submit Review" button.

Fig 16 – Items Details Page (Booking Request part in items details page)

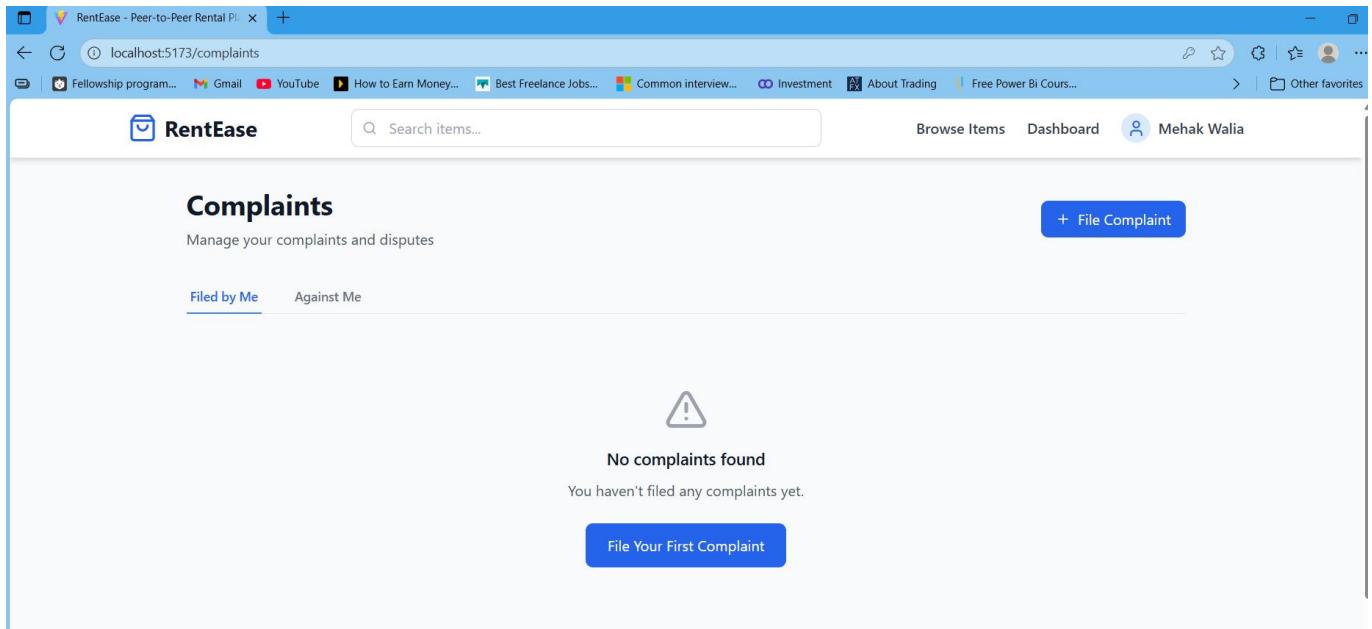


Fig 17 – Complaints Page

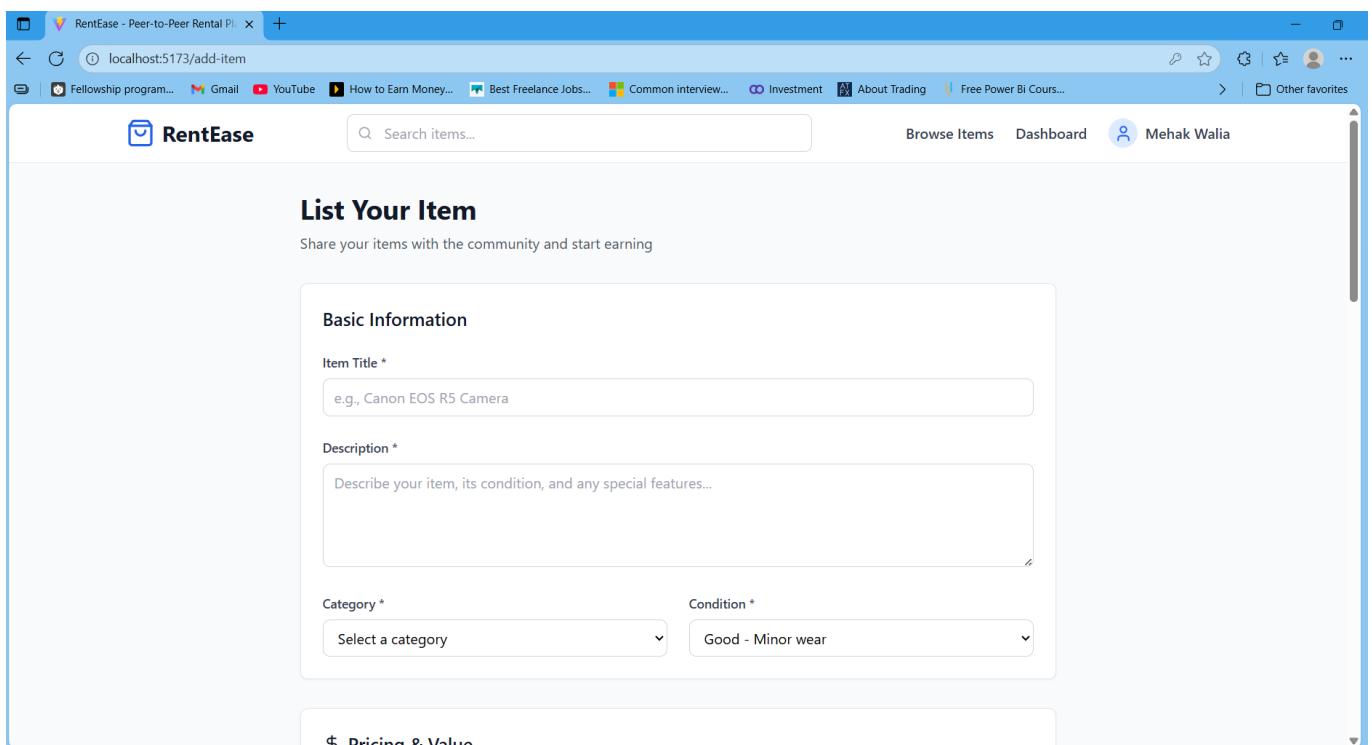


Fig 18 – Add Items Page

City *: Mumbai

State *: Maharashtra

ZIP Code *: 400001

Availability

Available From: dd-mm-yyyy

Available Until: dd-mm-yyyy

Leave empty if available indefinitely

Terms & Conditions

- You confirm that you own this item and have the right to rent it
- The item is in the condition described and images are accurate
- You agree to RentEase's terms of service and rental policies
- Platform commission: 5% of rental amount

Fig 19 – Add Items Page (Bottom Part)

Booking Type	Item Description	Duration	Cost	Lender/Borrower
Lending	Boult Speakers	25/10/2025 - 28/10/2025 3 days	₹2,337	Maridul Walia (Borrower)
Borrowing	Barack Obama Portrait	24/10/2025 - 26/10/2025 2 days	₹1,170	Maridul Walia (Lender)

Fig 20 - My Bookings Page

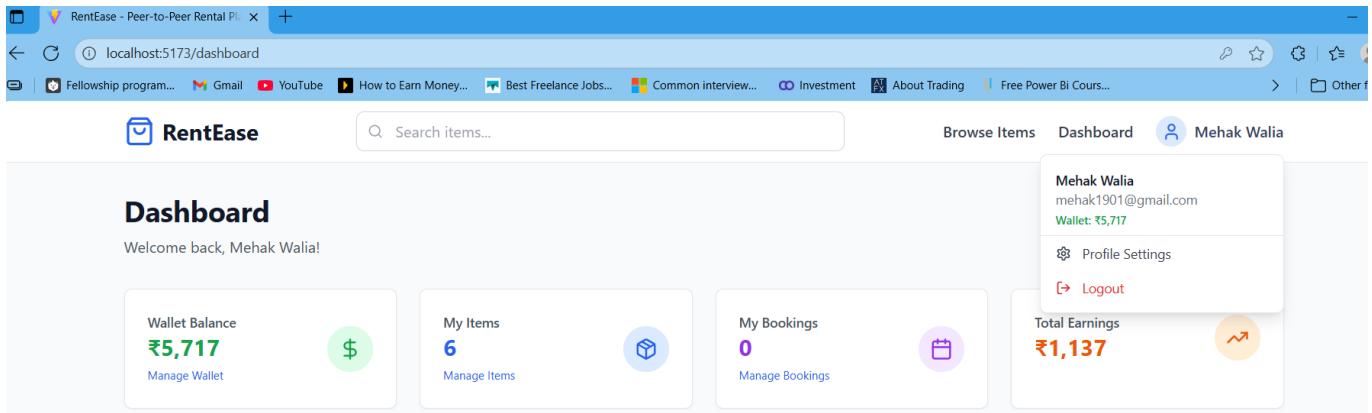


Fig 21 – Profile Management

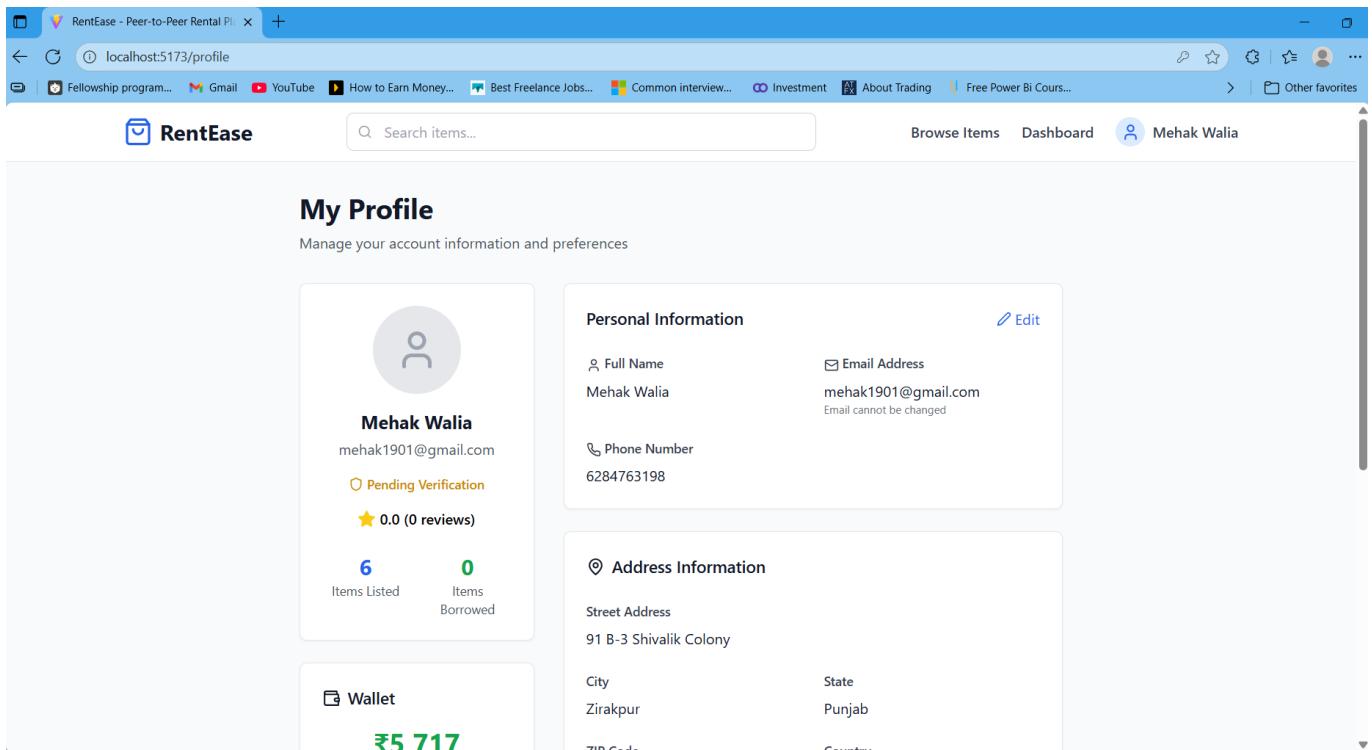


Fig 22 – My Profile

The screenshot shows the RentEase profile page for a user named Mehak Walia. The top navigation bar includes links for Fellowship program..., Gmail, YouTube, How to Earn Money..., Best Freelance Jobs..., Common interview..., Investment, About Trading, and Free Power Bi Cours... The main content area displays personal information, address details, and a wallet summary.

Personal Information:

- Full Name: Mehak Walia
- Email Address: mehak1901@gmail.com (Email cannot be changed)
- Phone Number: 6284763198

Address Information:

- Street Address: 91 B-3 Shivalik Colony
- City: Zirakpur
- State: Punjab
- ZIP Code: 140273
- Country: India

Wallet:

- Available Balance: ₹5,717
- Total Earnings: ₹1,137

Security:

- Change Password

Fig 23 – Profile Page

The screenshot shows the RentEase profile page for an Admin account. The top navigation bar includes links for Fellowship program..., Gmail, YouTube, How to Earn Money..., Best Freelance Jobs..., Common interview..., Investment, About Trading, and Free Power Bi Cours... The main content area displays personal information, address details, and a wallet summary. A dropdown menu for the Admin user is open, showing options: Profile Settings, Admin Panel, and Logout.

Personal Information:

- Full Name: Admin
- Email Address: admin@rentease.com (Email cannot be changed)
- Phone Number: 9897762317

Address Information:

- Street Address: Bandra West
- City: Mumbai
- State: Maharashtra
- ZIP Code: 400051
- Country: India

Logout:

Fig 24 – Profile Page (Admin Panel)

Admin Panel

Admin Dashboard
Overview of platform statistics and activities

- Total Users: 4
- Total Items: 24
- Active Bookings: 0
- Platform Revenue: ₹200

Recent Bookings

- Cricket Bat: Ramya Padala → Maridul Walia (completed)
- Boult Speakers: Maridul Walia → Mehak Walia (completed)
- Power Bank: Maridul Walia → Lakshya Ashwini (completed)
- Barack Obama Portrait: Mehak Walia → Maridul Walia (completed)

Recent Complaints

Fig 25 – Admin Panel (Dashboard Tab)

Admin Panel

Users Management
Manage platform users and their accounts

USER	CONTACT	STATS	STATUS	VERIFICATION	ACTIONS
Mehak Walia ID: 1f4fe8	mehak1901@gmail.com 6284763198	Items: 6 Rating: 0.0 (0)	Active	Unverified	Verify Suspend
Maridul Walia ID: a9bca8	maridul1897@gmail.com 8167630497	Items: 6 Rating: 0.0 (0)	Active	Unverified	Verify Suspend
Ramya Padala ID: a80360	ramya2587@gmail.com 1029384756	Items: 6 Rating: 0.0 (0)	Active	Verified	Unverify Suspend
Lakshya Ashwini ID: efbfdf	lakshya1883@gmail.com 9871234560	Items: 6 Rating: 0.0 (0)	Active	Verified	Unverify Suspend

Fig 26 – Admin Panel (User Management Tab)

The screenshot shows the RentEase Admin Panel with the 'Complaints Management' tab selected. The left sidebar has 'Admin Panel' and links to 'Dashboard', 'Users', 'Complaints' (which is highlighted in blue), 'Reviews', and 'Settings'. The main area title is 'Complaints Management' with the subtitle 'Review and resolve user complaints'. It features two filter dropdowns: 'Status Filter' set to 'All Statuses' and 'Priority Filter' set to 'All Priorities'. Below the filters is a table header with columns: COMPLAINT, PARTIES, PRIORITY, STATUS, DATE, and ACTIONS. A vertical scrollbar is visible on the right side of the main content area.

Fig 27 – Admin Panel (Complaints Management Tab)

The screenshot shows the RentEase Admin Panel with the 'Reviews Management' tab selected. The left sidebar has 'Admin Panel' and links to 'Dashboard', 'Users', 'Complaints', 'Reviews' (which is highlighted in blue), and 'Settings'. The main area title is 'Reviews Management' with the subtitle 'Moderate reported reviews and ratings'. It displays three summary boxes: 'Reported Reviews' (0), 'Pending Review' (0), and 'Hidden Reviews' (0). Below these is a table header with columns: REVIEW, RATING, REVIEWER, REVIEWEE, ITEM, DATE, and ACTIONS. A large message box in the center states 'No Reported Reviews' and 'All reviews are currently clean.' A vertical scrollbar is visible on the right side of the main content area.

Fig 28 – Admin Panel (Reviews Management Tab)

The screenshot shows the RentEase Admin Panel with the 'Admin' tab selected. On the left, there's a sidebar titled 'Admin Panel' containing links for Dashboard, Users, Complaints, Reviews, and Settings. The main area is titled 'Admin Settings' with the subtitle 'Configure platform settings and preferences'. It contains two sections: 'Platform Settings' and 'Rental Settings'. In 'Platform Settings', there are fields for 'Platform Fee (%)' (20), 'New User Bonus (₹)' (100), 'Referral Bonus (₹)' (500), and 'Max Items Per User' (10). In 'Rental Settings', there are fields for 'Minimum Rental Days' (1) and 'Maximum Rental Days' (30). A footer at the bottom left shows the URL 'localhost:5173/admin/users'.

Fig 29 – Admin Panel (Admin Settings Tab)

This screenshot shows the same RentEase Admin Panel as Fig 29, but with a different set of settings visible. It includes three main sections: 'Notification Settings', 'Support Settings', and 'System Settings'. Under 'Notification Settings', there are checkboxes for 'Enable email notifications' (checked) and 'Enable SMS notifications' (unchecked). Under 'Support Settings', there are fields for 'Support Email' (support@rentease.com) and 'Support Phone' (+91-9876543210). Under 'System Settings', there is a checkbox for 'Maintenance Mode' (unchecked), a note about restricting access during maintenance, a 'Maintenance Message' field containing 'System is currently under maintenance. Please try again later.', and a note about the message displayed to users during maintenance.

Fig 30 – Admin Panel (Admin Settings Tab)

5. Conclusion

The development of RentEase, a MERN-stack-based rental management platform, demonstrates the potential of modern web technologies in addressing real-world challenges within the sharing economy. Traditional peer-to-peer rental systems often face issues related to trust, communication gaps, lack of documentation, and inefficient coordination. By digitalizing the entire rental lifecycle—from user verification and item listing to booking management, pickup tracking, return confirmation, and reviews—RentEase provides a structured and transparent ecosystem where users can confidently engage in short-term rentals.

The project successfully integrates multiple modules such as user authentication, wallet management, item listing, booking workflows, and administrative controls within a unified and intuitive interface. The use of MongoDB ensures flexible data handling, Express.js and Node.js deliver robust API performance, and React.js provides a responsive and interactive user experience. The admin panel further strengthens platform governance by enabling user verification, complaint resolution, system configuration, and monitoring of platform-wide activities.

Although the current version utilizes local file storage and lacks deployment and version control integration, the system's architecture is designed to support future enhancements, including cloud-based file storage, containerized deployment using Docker, and hosting on scalable cloud platforms such as AWS or DigitalOcean. Additional features like payment gateway integration, automated security deposit handling, in-app notifications, and advanced analytics can be incorporated to further strengthen user engagement and operational efficiency.

Overall, RentEase serves as a functional and scalable prototype that showcases the viability of a digital rental marketplace. It addresses the core challenges of trust, workflow management, and transparency while providing a user-friendly platform for both item owners and renters. This project not only fulfills academic requirements but also has the potential to evolve into a market-ready application with further refinement, security hardening, and real-world deployment.

6. References

- Node.js Documentation – <https://nodejs.org>
- Express.js Official Guide – <https://expressjs.com>
- React.js Official Documentation – <https://react.dev>
- MongoDB Documentation – <https://www.mongodb.com/docs>
- Mongoose ODM Guide – <https://mongoosejs.com>
- Multer File Upload Documentation – [GitHub - expressjs/multer: Node.js middleware for handling `multipart/form-data`.](https://github.com/expressjs/multer#readme)
- Axios HTTP Client – <https://axios-http.com>